AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A process for converting a conventionally coded computer application program into a data set suitable for streamed delivery across a network from a server to a client in a computer environment, comprising the steps of:

providing installation monitoring means for monitoring an installation process of said conventionally coded application program on a local computer system;

wherein said installation monitoring means gathers modification information including ~~monitors~~ system registry modifications that said installation process makes to certain file paths in a system registry of said local computer system;

deceiving said client based on said gathered modification information ~~parameterizing said system registry modifications~~ by replacing certain of said file paths in said system registry modifications to re-direct requests for reading said system registry to a registry spoofer ~~with parameters that are recognizable by said client~~; and

providing data set creation means for processing said ~~parameterized system registry modifications~~ modification information for converting said application program into ~~to create~~ a data set suitable for deceiving said client into allowing streaming ~~said parameterized system registry modifications over said network~~ of bits of said data set over said network to said client such that said application program is capable of beginning execution on said client prior to downloading all of said application program.

2. (Original) The process of claim 1, wherein said data set creation means creates a runtime data set, said runtime data set consists of all regular application files and directories containing information about said regular application files.

3. (Original) The process of claim 2, wherein said data set creation means creates an initialization data set that is the first set of data streamed from said server to said client, said initialization data set prepares said client for streaming of said runtime data set.

4. (Original) The process of claim 2, wherein said directories contain lists of file names, file numbers, and the metadata associated with the files in a particular directory.

5. (Original) The process of claim 1, wherein said data set creation means creates a versioning table that contains a list of root file numbers and version numbers for tracking application patches and upgrades, and wherein each entry in said versioning table corresponds to one patch level of an application with a corresponding new root directory.

6. (Original) The process of claim 5, wherein said versioning table is sent to said client by said server, said client compares said versioning table with said client's root file number for the particular application program to find the necessary files required for a software upgrade or patch.

7. (Currently amended) The process of claim 1, further comprising the step of: providing a user interface that allows an operator to examine all changes made to said local computer system during said installation process and to edit said modification information ~~system modification data and said file modification data~~.

8. (Original) The process of claim 1, wherein said installation monitoring means monitors said application program as it runs and is being configured for a particular working environment on said local computer system and records common configurations of said application program thereby allowing said common configurations to be automatically duplicated on other client machines.

9. (Previously presented) The process of claim 1, further comprising program profiling means for capturing a sequence of file blocks being accessed during normal execution of said application program.

10. (Original) The process of claim 9, wherein said sequence of file blocks is used to pre-cache frequently used blocks on said client before said application program is first used by a user.

11. (Original) The process of claim 9, wherein said sequence of file blocks is used to optimize large directories of files on said client for faster file accesses.

12. (Original) The process of claim 9, wherein said sequence of file blocks is tied to specific user input and wherein said client pre-fetches file blocks based on user input to said application program.

13. (Currently amended) The process of claim 1, wherein said installation monitoring means records ~~the~~ a state of said local computer system before said installation process begins to give a more accurate picture of any modifications that are observed by said installation monitoring means.

14. (Currently Amended) An apparatus for converting a conventionally coded computer application program into a data set suitable for streamed delivery across a network from a server to a client in a computer environment, comprising:

installation monitoring module that monitors the installation process of said conventionally coded application program on a local computer system;

wherein said installation monitoring module gathers modification information including ~~that monitors~~ system registry modifications that said installation process makes to certain file paths in the system registry of said local computer system;

a module that deceives said client ~~parameterizes said system registry modifications~~ by replacing certain of said file paths to re-direct requests for reading said system registry to a registry spoofer ~~with parameters that are recognizable by said client~~; and

data set creation module that processes said ~~parameterized system registry modifications~~ modification information for converting said application program into ~~to create~~ a data set suitable for deceiving said client into allowing streaming ~~said parameterized system registry modifications over said network~~ of bits of said data set over said network to said client such that said application program is capable of beginning execution on said client prior to downloading all of said application program.

15. (Previously presented) The apparatus of claim 14, wherein said data set creation module creates a runtime data set, said runtime data set consists of all regular application files and directories containing information about said regular application files.

16. (Previously presented) The apparatus of claim 15, wherein said data set creation module creates an initialization data set that is the first set of data streamed from said server to said client, said initialization data set prepares said client for streaming of said runtime data set.

17. (Original) The apparatus of claim 15, wherein said directories contain lists of file names, file numbers, and the metadata associated with the files in a particular directory.

18. (Previously presented) The apparatus of claim 14, wherein said data set creation module creates a versioning table that contains a list of root file numbers and version numbers for tracking application patches and upgrades, and wherein each entry in said versioning table corresponds to one patch level of an application with a corresponding new root directory.

19. (Original) The apparatus of claim 18, wherein said versioning table is sent to said client by said server, said client compares said versioning table with said client's root file number for the particular application program to find the necessary files required for a software upgrade or patch.

20. (Currently amended) The apparatus of claim 14, further comprising: a user interface that allows an operator to examine all changes made to said local computer system during said installation process and to edit said ~~system modification data and said file modification data~~ modification information.

21. (Previously presented) The apparatus of claim 14, wherein said installation monitoring module monitors said application program as it runs and is being configured for

a particular working environment on said local computer system and records common configurations of said application program thereby allowing said common configurations to be automatically duplicated on other client machines.

22. (Previously presented) The apparatus of claim 14, further comprising: program profiling module that captures sequence of file blocks being accessed during normal execution of said application program.

23. (Original) The apparatus of claim 22, wherein said sequence of file blocks is used to pre-cache frequently used blocks on said client before said application program is first used by a user.

24. (Original) The apparatus of claim 22, wherein said sequence of file blocks is used to optimize large directories of files on said client for faster file accesses.

25. (Original) The apparatus of claim 22, wherein said sequence of file blocks is tied to specific user input and wherein said client pre-fetches file blocks based on user input to said application program.

26. (Currently amended) The apparatus of claim 14, wherein said installation monitoring module records ~~the~~ a state of said local computer system before said installation process begins to give a more accurate picture of any modifications that are observed by said installation monitoring.

27. (Currently Amended) A program storage medium readable by a computer, tangibly embodying a program of instructions executable by the computer to perform method steps for converting a conventionally coded computer application program into a

data set suitable for streamed delivery across a network from a server to a client in a computer environment, comprising:

monitoring the installation process of said conventionally coded application program on a local computer system;

gathering modification information including ~~monitoring the~~ system registry modifications that said installation process makes to certain file paths in the system registry of said local computer system;

deceiving said client based on said gathered modification information ~~parameterizing said system registry modifications~~ by replacing certain of said file paths ~~with parameters that are recognizable by said client~~; and

processing said ~~parameterized system registry modifications~~ modification information for converting said application program into a form for deceiving said client into allowing streaming of bits of said converted application program over said network to said client such that said application program is capable of beginning execution on said client prior to downloading all of said application program.

28. (Previously presented) The method of claim 27, further comprising creating a runtime data set, said runtime data set comprises regular application files and directories containing information about said regular application files.

29. (Previously presented) The method of claim 28, further comprising creating an initialization data set that is the first set of data streamed from said server to said client, said initialization data set prepares said client for streaming of said runtime data set.

30. (Original) The method of claim 28, wherein said directories contain lists of file names, file numbers, and the metadata associated with the files in a particular directory.

31. (Previously presented) The method of claim 27, wherein further comprising creating a versioning table that contains a list of root file numbers and version numbers for tracking application patches and upgrades, and wherein each entry in said versioning table corresponds to one patch level of an application with a corresponding new root directory.

32. (Original) The method of claim 31, wherein said versioning table is sent to said client by said server, said client compares said versioning table with said client's root file number for the particular application program to find the necessary files required for a software upgrade or patch.

33. (Previously presented) The method of claim 27, further comprising providing a user interface that allows an operator to examine changes made to said local computer system during said installation process and to edit said system modification data and said file modification data.

34. (Previously presented) The method of claim 27, further comprising monitoring said application program as it runs and is being configured for a particular working environment on said local computer system and records common configurations of said application program thereby allowing said common configurations to be automatically duplicated on other client machines.

35. (Previously presented) The method of claim 27, further comprising capturing a sequence of file blocks being accessed during normal execution of said application program.

36. (Original) The method of claim 35, wherein said sequence of file blocks is used to pre-cache frequently used blocks on said client before said application program is first used by a user.

37. (Original) The method of claim 35, wherein said sequence of file blocks is used to optimize large directories of files on said client for faster file accesses.

38. (Original) The method of claim 35, wherein said sequence of file blocks is tied to specific user input and wherein said client pre-fetches file blocks based on user input to said application program.

39. (Currently Amended) The method of claim 27, wherein said installation monitoring means records ~~the~~ a state of said local computer system before said installation process begins to give a more accurate picture of any modifications that are observed by said installation monitoring means.

40. (Currently amended) A method for converting an application program into a data set suitable for streamed delivery across a network from a server to a client in a computer environment, the method comprising:

monitoring an installation process of an application program on a local computer system;

~~monitoring system registry~~ gathering modification information including information on modifications that said installation process makes to certain file paths in a system registry of said local computer system;

deceiving said client based on said gathered modification information ~~parameterizing said system registry modifications~~ by replacing certain of said file

paths in said system registry modifications to re-direct requests for reading said system registry to a registry spoofer ~~with parameters that are recognizable by a client~~; and

processing ~~said parameterized system registry modifications~~ said modification information for converting said application program into ~~to create~~ a data set suitable for deceiving said client into allowing streaming ~~said parameterized system registry modifications~~ of bits of said data set over said network to said client such that said application program is capable of beginning execution on said client prior to downloading all of said application program.

41. (Previously presented) The method of claim 40, further comprising creating a runtime data set, said runtime data set consists of regular application files and directories containing information about said regular application files.

42. (Previously presented) The method of claim 41, further comprising creating an initialization data set that is the first set of data streamed from said server to said client, said initialization data set prepares said client for streaming of said runtime data set.

43. (Previously presented) The method of claim 41, wherein said directories contain lists of file names, file numbers, and the metadata associated with the files in a particular directory.

44. (Previously presented) The method of claim 40, further comprising creating a versioning table that contains a list of root file numbers and version numbers for tracking application patches and upgrades, and wherein each entry in said versioning table corresponds to one patch level of an application with a corresponding new root directory.

45. (Previously presented) The method of claim 44, wherein said versioning table is sent to said client by said server, said client compares said versioning table with said client's root file number for the particular application program to find the necessary files required for a software upgrade or patch.

46. (Previously presented) The method of claim, 40, further comprising providing a user interface that allows an operator to examine all changes made to said local computer system during said installation process and to edit said system modification data and said file modification data.

47. (Previously presented) The method of claim 40, further comprising monitoring said application program as it runs and is being configured for a particular working environment on said local computer system and records common configurations of said application program thereby allowing said common configurations to be automatically duplicated on other client machines.

48. (Previously presented) The method of claim 40, further comprising capturing a sequence of file blocks being accessed during normal execution of said application program.

49. (Previously presented) The method of claim 48, wherein said sequence of file blocks is used to pre-cache frequently used blocks on said client before said application program is first used by a user.

50. (Previously presented) The method of claim 48, wherein said sequence of file blocks is used to optimize large directories of files on said client for faster file accesses.

51. (Previously presented) The method of claim 48, wherein said sequence of file blocks is tied to specific user input and wherein said client pre-fetches file blocks based on user input to said application program.

52. (Currently amended) The method of claim 40, further comprising recording ~~the~~ a state of said local computer system before said installation process begins.